HyperQuery: A Framework for Higher Order Link Prediction

Anonymous Author(s)
Affiliation
Address
email

A Additional Information for Hyperedge Prediction

Table 1 shows the mean test auc and standard deviation over 10 different train-test splits. We compare *HyperQuery* with two other state of the art baseline methods. We find that our approach outperforms the other methods by a large margin, with high confidence.

Augmented HyperQuery Classifier (Baseline): As discussed in the main paper, we evaluate the baseline that treats hyperedge prediction problem as a classification task where we add an extra label in our dataset that represents the negative hyperedges (hyperedges that do not exist) and use our classification framework without community detection.

HyperQuery-minmax: The best performing hyperedge prediction formulation presented in this paper. We use operator *minmax* for the Ω aggregation function. We use 16 communities for our community detection step.

NHP-minmax: GCN based hyperedge prediction that can be used for directed and undirected hypergraphs. We used the minmax operator for this approach as well.

HyperSAGNN: A self attention approach for hyperedge prediction that can only be used for undirected hypergraphs. We used random-walk basd approach.

Comparing the performance of *HyperQuery* with the baselines clearly demonstrates the effectiveness of leveraging community detection algorithms to improve prediction accuracy. Our explanation for this is that community detection algorithms minimize global objectives, such as approximations of the NP-Hard multiway cut problem, whereas existing algorithms rely on purely local approaches like random walks and graph convolutions to learn salient features. Our results demonstrate that partitioning with community detection can be an extremely powerful tool when one sets out to learn useful representations of hypergraph structured data.

Table 1: Area Under Curve (AUC) scores for hyperedge prediction.

	IAF1260B AUC	**IJO1366** AUC	**USPTO** AUC	**DBLP** AUC
BASELINE	62.4 ±.4	60.4±1.	65.5±.1	65.1±1.
HYPERQUERY-MINMAX	**72.2** ±.5	**68.5**±1.	**75.7**±.1	**72.0**±1.
NHP-MINMAX	64.3 ±3.	63.2 ±2.	74.2 ±2.	69.2 ±2.
HYPER-SAGNN	60.1 ±1.	56.3 ±2.	67.1 ±1.	65.2 ±1.

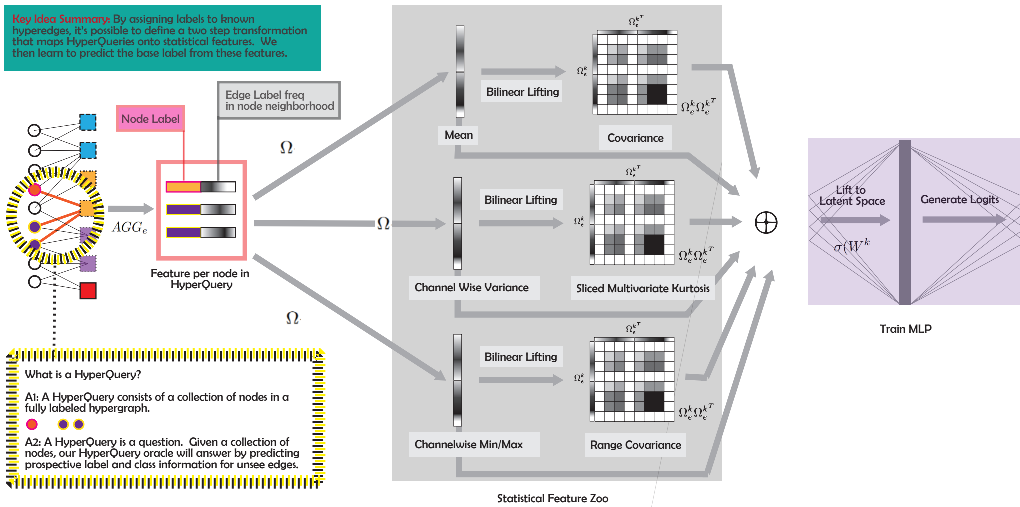

Figure 1: In the body of the main paper, we evaluate our HyperQuery pipeline trained with different sets of statistical features. In particular, HYPERQUERY-MEAN computes the coordinate wise mean of the node feature distribution, and after bilinear lifting this feature corresponds to the covariance matrix of this distribution. HYPERQUERY-VAR and HYPERQUERY-MINMAX measure the closely related coordinate-wise sample range and sample variance, which perform somewhat similarly in practice. The bilinear pooling operation in the case of variance can be thought of as computing a particularly relevant slice of the full 4^{th} order multivariate kurtosis tensor. While we have had trouble finding an analogous measure in the literature, we find that using such "Sliced Multivariate Kurtosis" features can improve hyperedge prediction accuracy, particularly on larger datasets. Our explanation of this phenomena is that hypergraphs encode higher order relationships between nodes, and thus capturing higher order statistics is essential to building reliable models for hyperedge prediction and classification.

B Illuminating HyperQuery Representations

While hyperedge prediction has been studied in a number of settings, our approach of using community detection algorithms to define categorical labels on hyperedges sets HyperQuery apart from prior art, both in terms of methodology and in terms of performance.

In figure 1 we illustrate the zoo of statistical features that we evaluated. Though concatenating these features together can improve performance of our pipeline, however in our paper we focused on evaluating each statistical feature separately in order to more clearly illustrate their relative effectiveness for our benchmark tasks. In future work, it would be interesting to explore learned approaches to this feature generation step as well.

C Time complexity

We analyze the time complexity of our framework (*i.e.*, Equations 9, 10,and 11 in the main paper in terms of the size of the hypergraph, assuming the number of labels is independent of the size of the hypergraph. Let m denote the number of hyperedges and n denote the number of nodes. Let $deg(v_i)$ denote the degree of node v_i and let Δ denote $max_{1\leq i\leq n}deg(v_i)$. For Equation 9, our framework takes $O(n \cdot \Delta)$ time.

For Equation 10, let $deg(e_i)$ denote the degree of hyperedge e_i and let Δ denote $max_{1\leq i\leq m}deg(e_i)$. Equation 10 takes $O(m \cdot \Delta)$ time. Finally, the time complexity of Equation 11 is constant since the number of labels is independent of the size of the graph. Overall, our framework takes $O(m \cdot \Delta) + O(n \cdot \Delta)$.

References

[1] Alexander, J.A. & Mozer, M.C. (1995) Template-based algorithms for connectionist rule extraction. In G. Tesauro, D.S. Touretzky and T.K. Leen (eds.), *Advances in Neural Information Processing Systems 7*, pp. 609–616. Cambridge, MA: MIT Press.

[2] Bower, J.M. & Beeman, D. (1995) *The Book of GENESIS: Exploring Realistic Neural Models with the GEneral NEural SImulation System.* New York: TELOS/Springer–Verlag.

[3] Hasselmo, M.E., Schnell, E. & Barkai, E. (1995) Dynamics of learning and recall at excitatory recurrent synapses and cholinergic modulation in rat hippocampal region CA3. *Journal of Neuroscience* **15**(7):5249-5262.